



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: Marco Antonio Quintana

Asignatura: EDA-1

Grupo: 17

No de Práctica(s): 9

Integrante(s): Menes Pacheco Sebastián Efraín

*No. de Equipo de
cómputo empleado:* 27

No. de Lista o Brigada: 24

Semestre: 2020-2

Fecha de entrega: 5/04/20

Observaciones:

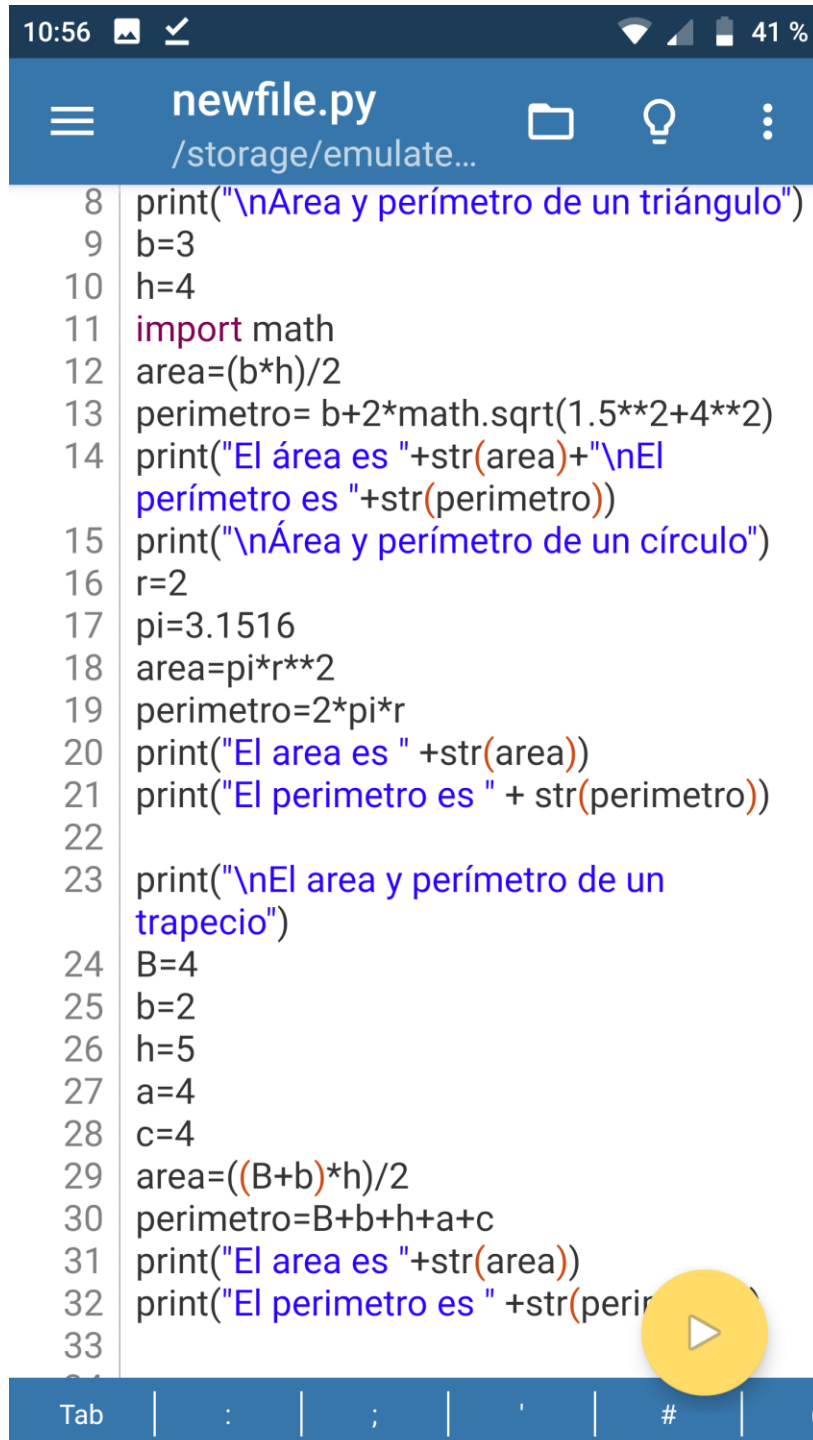
CALIFICACIÓN: _____

Introducción

En la práctica 9 correspondiente a introducción a Python, se nos enseñara lo básico empezando desde imprimir caracteres, concatenarlos, cambiar su orden. A continuación se verá operadores aritméticos, suma resta, multiplicación y division. Veremos listas, diccionarios y variables globales.

Desarrollo

Área y perímetro de rectángulo, triángulo, círculo y trapecio.



```
10:56 41 %
newfile.py
/storage/emulate...

8 print("\nÁrea y perímetro de un triángulo")
9 b=3
10 h=4
11 import math
12 area=(b*h)/2
13 perimetro= b+2*math.sqrt(1.5**2+4**2)
14 print("El área es "+str(area)+"\nEl
   perímetro es "+str(perimetro))
15 print("\nÁrea y perímetro de un círculo")
16 r=2
17 pi=3.1516
18 area=pi*r**2
19 perimetro=2*pi*r
20 print("El area es "+str(area))
21 print("El perimetro es " + str(perimetro))
22
23 print("\nEl area y perímetro de un
   trapecio")
24 B=4
25 b=2
26 h=5
27 a=4
28 c=4
29 area=((B+b)*h)/2
30 perimetro=B+b+h+a+c
31 print("El area es "+str(area))
32 print("El perimetro es "+str(perimetro))
33
```

10:56



41 %



TAB



Area y perímetro de un rectángulo

El area es 8

El perímetro es 12

Area y perímetro de un triángulo

El área es 6.0

El perímetro es 11.54400374531753

Área y perímetro de un círculo

El area es 12.6064

El perimetro es 12.6064

El area y perímetro de un trapecio

El area es 15.0

El perimetro es 19

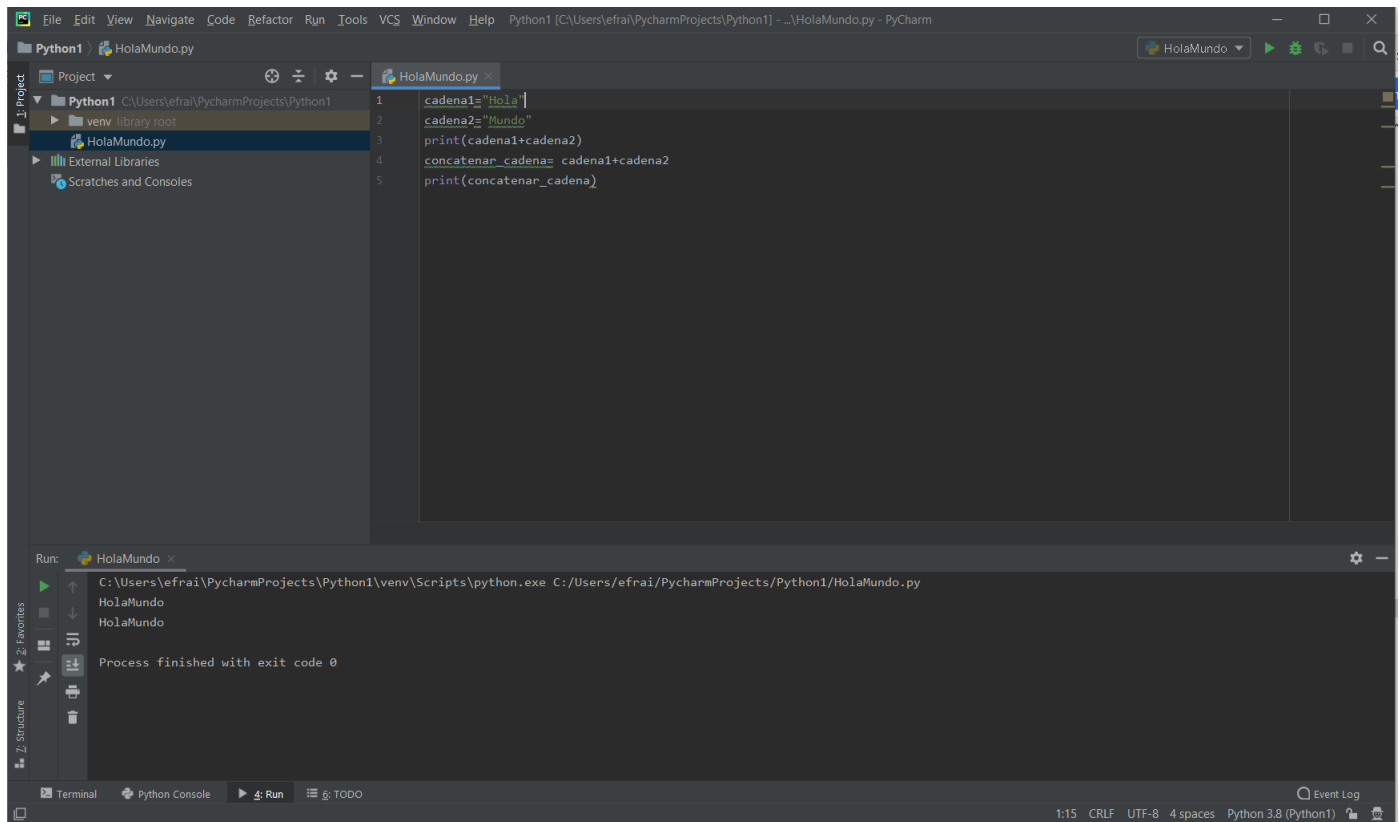
[Program finished]



```
1 print("Area y perímetro de un rectángulo")
2 b=4
3 h=2
4 area=b*h
5 perimetro=b*2+h*2
6 print("El area es "+ str(area)+"\nEl
  perímetro es "+ str(perimetro))
7
8 print("\nArea y perímetro de un triángulo")
9 b=3
10 h=4
11 import math
12 area=(b*h)/2
13 perimetro= b+2*math.sqrt(1.5**2+4**2)
14 print("El área es "+str(area)+"\nEl
  perímetro es "+str(perimetro))
15 print("\nÁrea y perímetro de un círculo")
16 r=2
17 pi=3.1516
18 area=pi*r**2
19 perimetro=2*pi*r
20 print("El area es "+str(area))
21 print("El perimetro es " + str(perimetro))
22
23 print("\nEl area y perímetro de un
  trapecio")
24 B=4
25 b=2
```



Hola mundo usando cadenas y cadenas concatenadas



The screenshot shows the PyCharm IDE with a project named 'Python1'. The file explorer on the left shows the project structure, including a 'venv' directory and a file named 'HolaMundo.py'. The main editor window displays the code for 'HolaMundo.py':

```
1 cadena1="Hola"
2 cadena2="Mundo"
3 print(cadena1+cadena2)
4 concatenar_cadena= cadena1+cadena2
5 print(concatenar_cadena)
```

Below the editor, the 'Run' window shows the execution of the script. The command used is:

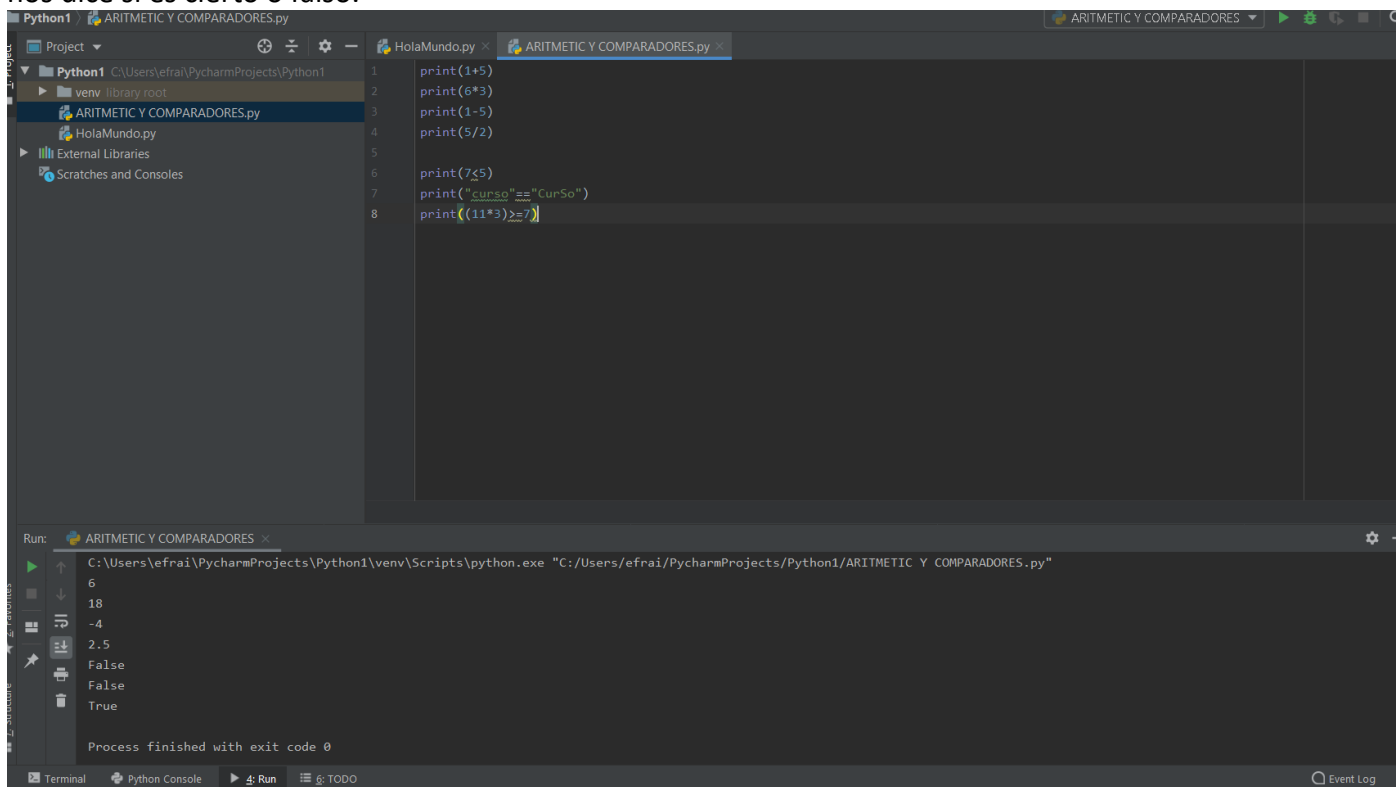
```
C:\Users\efrai\PycharmProjects\Python1\venv\Scripts\python.exe C:/Users/efrai/PycharmProjects/Python1/HolaMundo.py
```

The output of the script is:

```
HolaMundo
HolaMundo
```

The process finished with exit code 0.

Aquí se hace uso de los operadores aritmeticos y las desigualdades donde podemos ver que el programa nos dice si es cierto o falso.



The screenshot shows the PyCharm IDE with a project named 'Python1'. The file explorer on the left shows the project structure, including a 'venv' directory and a file named 'ARITMETIC Y COMPARADORES.py'. The main editor window displays the code for 'ARITMETIC Y COMPARADORES.py':

```
1 print(1+5)
2 print(6*3)
3 print(1-5)
4 print(5/2)
5
6 print(7<5)
7 print("curso"=="CurSo")
8 print((11*3)>=7)
```

Below the editor, the 'Run' window shows the execution of the script. The command used is:

```
C:\Users\efrai\PycharmProjects\Python1\venv\Scripts\python.exe "C:/Users/efrai/PycharmProjects/Python1/ARITMETIC Y COMPARADORES.py"
```

The output of the script is:

```
6
18
-4
2.5
False
False
True
```

The process finished with exit code 0.

Las listas y listas anidadas

The screenshot shows the PyCharm IDE with a project named 'Python1'. The file explorer on the left shows the project structure, including a 'venv' directory and several Python files. The main editor displays the code for 'Listas.py':

```
1 #Declaracion de listas
2 listas_diasDelMes=[28,27,26,25]
3 print(listas_diasDelMes)
4 print(listas_diasDelMes[0])
5 print(listas_diasDelMes[2])
6
7 #Declaracion de listas anidadas
8 listas_anidadas=[["cero", 0], ["uno", 1], ["dos", 2], ["X", 3]]
9 print(listas_anidadas)
10 print(listas_anidadas[0])
11 print(listas_anidadas[2][0])
12 #se cambia el valor de uno de los elementos de la listas
13 listas_anidadas[3][0]="5"
14 print(listas_anidadas[3])
```

The Run console at the bottom shows the output of the script:

```
C:\Users\efrai\PycharmProjects\Python1\venv\Scripts\python.exe C:/Users/efrai/PycharmProjects/Python1/Listas.py
[28, 27, 26, 25]
28
26
[['cero', 0], ['uno', 1], ['dos', 2], ['X', 3]]
['cero', 0]
dos
['5', 3]
Process finished with exit code 0
```

Tuplas que son como listas pero no se pueden cambiar a diferencias de las listas

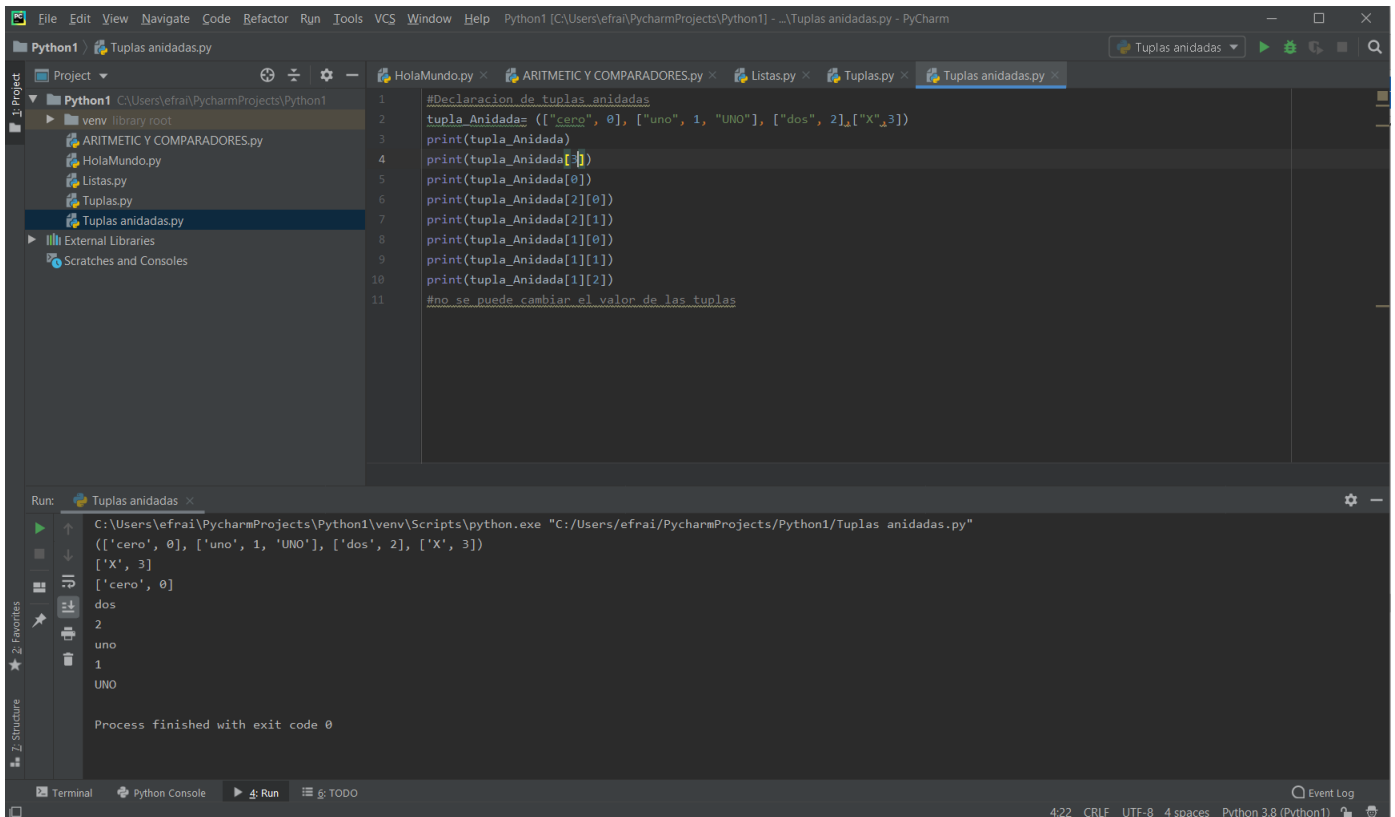
The screenshot shows the PyCharm IDE with a project named 'Python1'. The file explorer on the left shows the project structure, including a 'venv' directory and several Python files. The main editor displays the code for 'Tuplas.py':

```
1 #Declaración de tuplas
2 tupla_dias=(21,28,43,12,23,123,83)
3 print(tupla_dias)
4 print(tupla_dias[1])
5 print(tupla_dias[4])
6 print(tupla_dias[0])
```

The Run console at the bottom shows the output of the script:

```
C:\Users\efrai\PycharmProjects\Python1\venv\Scripts\python.exe C:/Users/efrai/PycharmProjects/Python1/Tuplas.py
(21, 28, 43, 12, 23, 123, 83)
28
23
21
Process finished with exit code 0
```

Las tuplas anidadas para hacer un conjunto de subconjuntos de elementos y poderlos cambiar



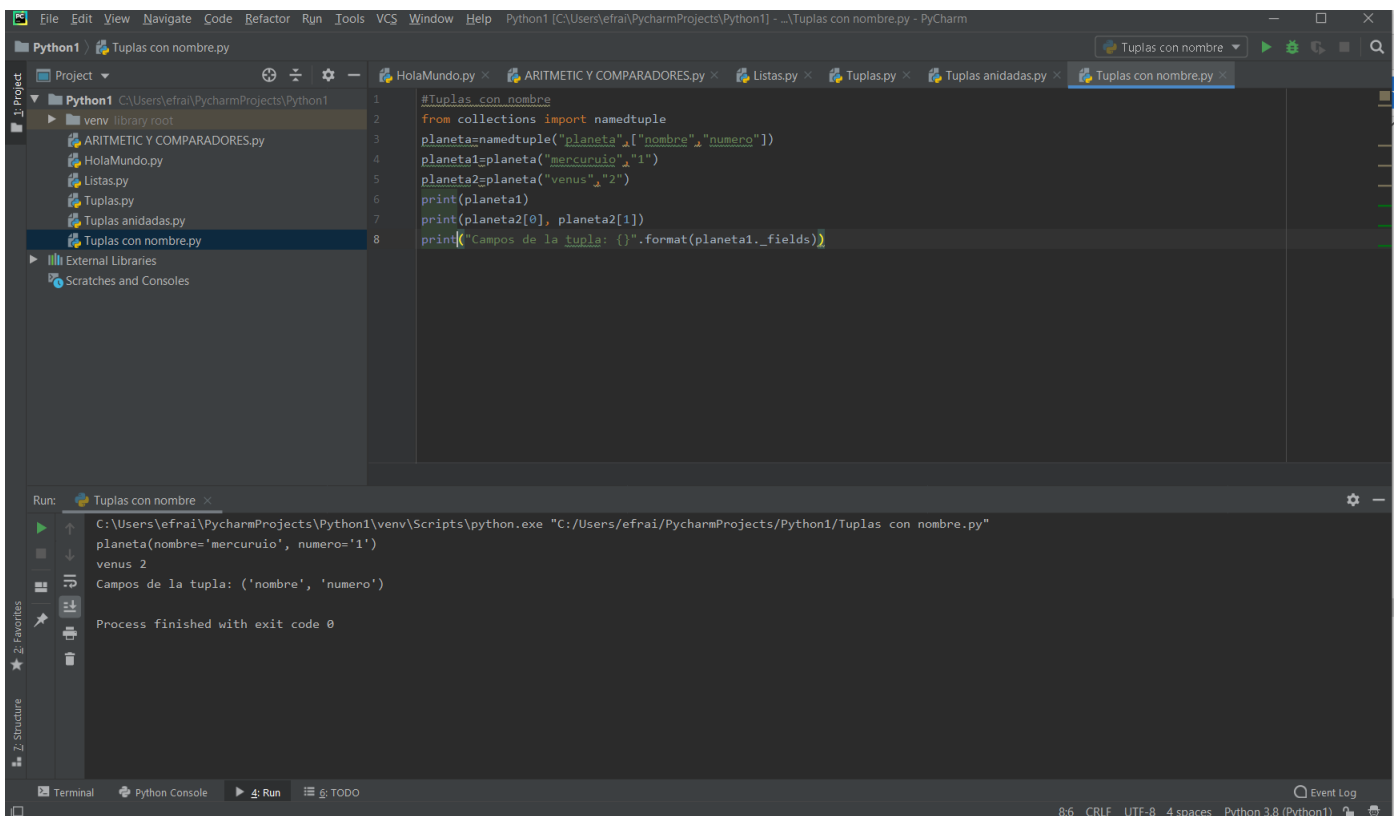
The screenshot shows the PyCharm IDE with a project named 'Python1'. The file explorer on the left shows a directory structure with files like 'ARITMETIC Y COMPARADORES.py', 'HolaMundo.py', 'Listas.py', 'Tuplas.py', and 'Tuplas anidadas.py'. The main editor displays the code in 'Tuplas anidadas.py':

```
1 #Declaracion de tuplas anidadas
2 tupla_Anidadas= (["cero", 0], ["uno", 1, "UNO"], ["dos", 2], ["X", 3])
3 print(tupla_Anidadas)
4 print(tupla_Anidadas[:1])
5 print(tupla_Anidadas[0])
6 print(tupla_Anidadas[2][0])
7 print(tupla_Anidadas[2][1])
8 print(tupla_Anidadas[1][0])
9 print(tupla_Anidadas[1][1])
10 print(tupla_Anidadas[1][2])
11 #no se puede cambiar el valor de las tuplas
```

The Run console at the bottom shows the output of the script:

```
Run: Tuplas anidadas
C:\Users\efrai\PycharmProjects\Python1\venv\Scripts\python.exe "C:/Users/efrai/PycharmProjects/Python1/Tuplas anidadas.py"
(['cero', 0], ['uno', 1, 'UNO'], ['dos', 2], ['X', 3])
['X', 3]
['cero', 0]
dos
2
uno
1
UNO
Process finished with exit code 0
```

Aquí se muestran las tuplas con nombre que es una lista la cual se pueden poner como subcategorias



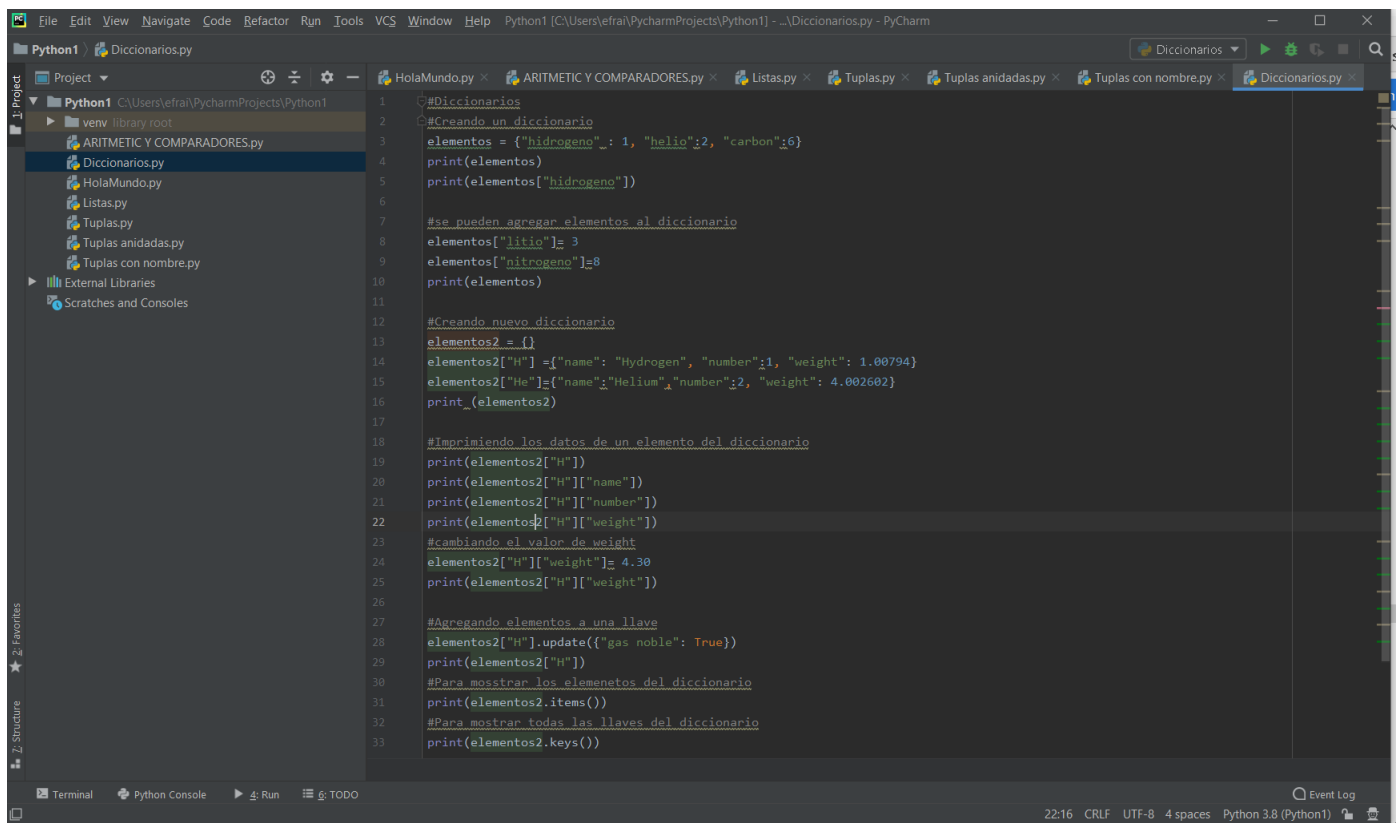
The screenshot shows the PyCharm IDE with a project named 'Python1'. The file explorer on the left shows a directory structure with files like 'ARITMETIC Y COMPARADORES.py', 'HolaMundo.py', 'Listas.py', 'Tuplas.py', 'Tuplas anidadas.py', and 'Tuplas con nombre.py'. The main editor displays the code in 'Tuplas con nombre.py':

```
1 #Tuplas con nombre
2 from collections import namedtuple
3 planeta=namedtuple("planeta",["nombre","numero"])
4 planeta1=planeta("mercurio",1)
5 planeta2=planeta("venus",2)
6 print(planeta1)
7 print(planeta2[0], planeta2[1])
8 print("Campos de la tupla: {}".format(planeta1._fields))
```

The Run console at the bottom shows the output of the script:

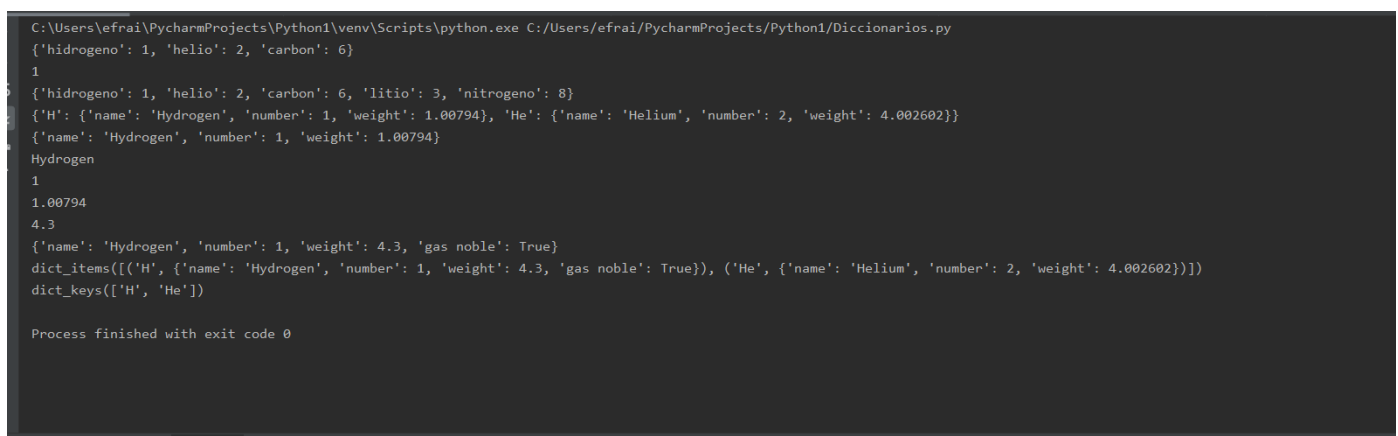
```
Run: Tuplas con nombre
C:\Users\efrai\PycharmProjects\Python1\venv\Scripts\python.exe "C:/Users/efrai/PycharmProjects/Python1/Tuplas con nombre.py"
planeta(nombre='mercurio', numero=1)
venus 2
Campos de la tupla: ('nombre', 'numero')
Process finished with exit code 0
```

Los diccionarios que nos sirven para hacer un listado de elementos con respectivas categorías como es el caso de los elementos que se nos muestra su nombre, número que ocupa y su peso. También se pueden modificar estos elementos, agregarlos o cambiarlos con respectivas funciones.



The screenshot shows the PyCharm IDE with a project named 'Python1'. The file explorer on the left shows a directory structure with files like 'Diccionarios.py', 'HolaMundo.py', 'Listas.py', 'Tuplas.py', 'Tuplas anidadas.py', and 'Tuplas con nombre.py'. The main editor window displays the 'Diccionarios.py' file with the following code:

```
1 #Diccionarios
2 #Creando un diccionario
3 elementos = {"hidrogeno": 1, "helio": 2, "carbon": 6}
4 print(elementos)
5 print(elementos["hidrogeno"])
6
7 #se pueden agregar elementos al diccionario
8 elementos["litio"] = 3
9 elementos["nitrogeno"] = 8
10 print(elementos)
11
12 #Creando nuevo diccionario
13 elementos2 = {}
14 elementos2["H"] = {"name": "Hydrogen", "number": 1, "weight": 1.00794}
15 elementos2["He"] = {"name": "Helium", "number": 2, "weight": 4.002602}
16 print(elementos2)
17
18 #Imprimiendo los datos de un elemento del diccionario
19 print(elementos2["H"])
20 print(elementos2["H"]["name"])
21 print(elementos2["H"]["number"])
22 print(elementos2["H"]["weight"])
23
24 #cambiando el valor de weight
25 elementos2["H"]["weight"] = 4.30
26 print(elementos2["H"]["weight"])
27
28 #Agregando elementos a una llave
29 elementos2["H"].update({"gas noble": True})
30 print(elementos2["H"])
31 #Para mostrar los elementos del diccionario
32 print(elementos2.items())
33 #Para mostrar todas las llaves del diccionario
34 print(elementos2.keys())
```

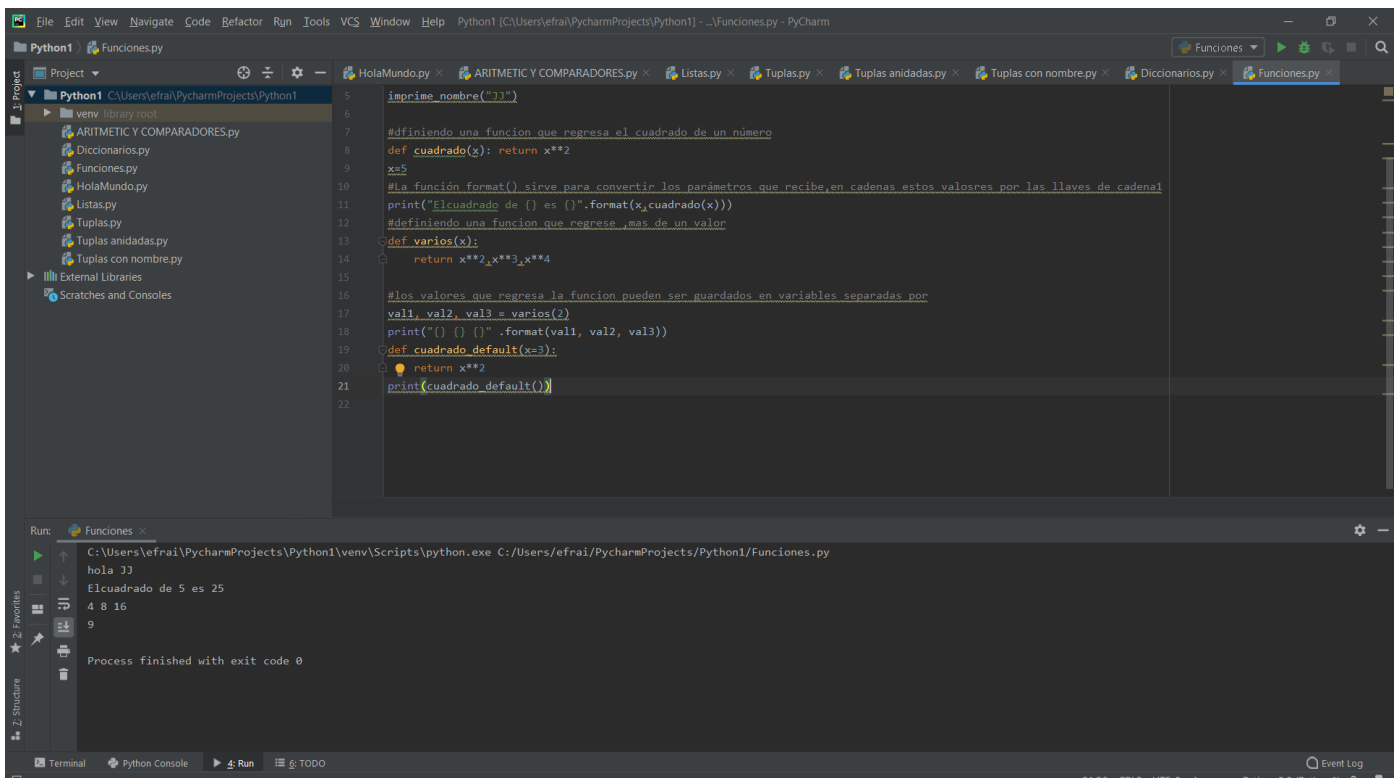


The screenshot shows a terminal window with the following output:

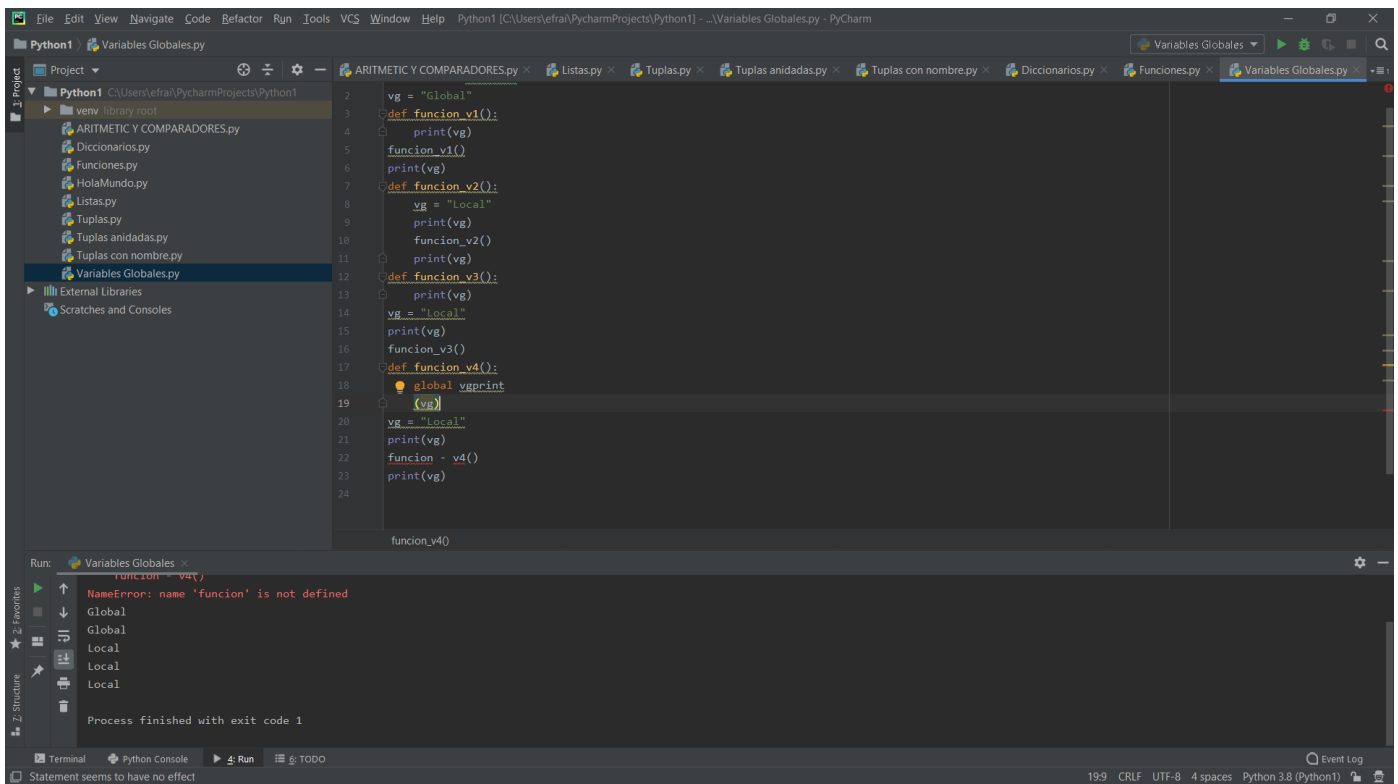
```
C:\Users\efrai\PycharmProjects\Python1\venv\Scripts\python.exe C:/Users/efrai/PycharmProjects/Python1/Diccionarios.py
{'hidrogeno': 1, 'helio': 2, 'carbon': 6}
1
{'hidrogeno': 1, 'helio': 2, 'carbon': 6, 'litio': 3, 'nitrogeno': 8}
{'H': {'name': 'Hydrogen', 'number': 1, 'weight': 1.00794}, 'He': {'name': 'Helium', 'number': 2, 'weight': 4.002602}}
{'name': 'Hydrogen', 'number': 1, 'weight': 1.00794}
Hydrogen
1
1.00794
4.3
{'name': 'Hydrogen', 'number': 1, 'weight': 4.3, 'gas noble': True}
dict_items([('H', {'name': 'Hydrogen', 'number': 1, 'weight': 4.3, 'gas noble': True}), ('He', {'name': 'Helium', 'number': 2, 'weight': 4.002602})])
dict_keys(['H', 'He'])

Process finished with exit code 0
```

Aquí se nos muestran las funciones



Las variables globales



Conclusión

En general se me hizo una práctica bastante amigable ya que como temenos conocimietos de lenguaje c es mucho más facil entender el lenguaje ya que es muy parecido en las funciones y hasta de alguna manera más simplificado. Cabe destacar que debemos entender bien las funciones y comandos necesarios para realizar las tareas o problemas que se nos presente. Es muy importante como se menciona en la práctica el orden que debe tener porque si no el programa como que no detecta la instrucción.

Biobliografía

<https://docs.python.org/3/tutorial/>

<https://www.jetbrains.com/es-es/pycharm/download/download-thanks.html?platform=windows&code=PCC>

<https://www.youtube.com/watch?reload=9&v=DAdRO6ByBoU>

<https://realpython.com/python-ides-code-editors-guide/#eclipse-pydev>